Arid Zone Journal of Engineering, Technology and Environment, December, 2017; Vol. 13(6):764-778

Print ISSN: 1596-2490, Electronic ISSN: 2545-5818, www.azojete.com.ng

AN INVESTIGATION INTO THE EFFECTIVENESS OF MACHINE LEARNING TECHNIQUES FOR INTRUSION DETECTION

E. G. Dada[1], J. S. Bassi[1] and O. O. Adekunle[2]

[1]*Department of Computer Engineering, Faculty of Engineering, University of Maiduguri, P.M.B 1069, Maiduguri, Borno State. Nigeria*

[2]*Department of Computer Science, Faculty of Science, National Open University of Nigeria (NOUN), 91, Cadastral Zone, Nnamdi Azikiwe Expressway, Jabi, Abuja, Nigeria.*

Correspondence e-mail: *gbengadada@unimaid.edu.ng*

Abstract

Attacks on computer systems are becoming progressively frequent. Many machine learning techniques have been developed in the bid to increase the effectiveness of intrusion detection systems (IDS). However, the sophistication of intrusion attacks on computer networks and the large size of dataset pose a serious challenge as they drastically reduce the effectiveness of the IDS. We do not propose any new algorithm in this paper. However, experiments were conducted to investigate the performance of six (6) machine learning techniques found in literature and how they can effectively detect intrusion activities on a network. This work examines how effective each algorithm under investigation handles intrusion events. In our experiment, the NSL-KDDTrain+ dataset was partitioned into training subgroups subject to the type of network protocol. Subsequent to this, extraneous and unneeded attributes are removed from each training subgroup. The effectiveness of the algorithms was evaluated. The experimental results show that the Logistic Model Tree Induction method is more effective in terms of (classification accuracy: 99.40%, F-measure: 0.991, false positive rate: 0.32%, precision: 98.90% and Receiver Operating Characteristics: 98.6%) compared to the other five machine learning techniques we investigated.

Keywords: Multiple Linear Perceptron, Support Vector Machine, Random Forests, Logistic Model Tree Induction, Decision Tree and Naïve Bayes.

1. Introduction

Intrusion Detection System (IDS) can be defined as the process of recognizing and countering mischievous activity aimed at computer system and networking resources (Hamdan, *et al.*, 2010) and (Liao, *et al.*, 2013). They are mechanisms (usually another computer) employed to monitor the activities going on in the network in order to discover malevolentor in appropriate incidents (Jatuphum, *et al.*, 2015). An IDS takes new input from sensors, scrutinizes these inputs and then swing into action (Kavitha and Suresh, 2012). Realising the fact that the cost of information processing and Internet accessibility is crashing, organizations to a greater extent are becoming susceptible to diversities of cyber vulnerabilities (Salma, *et al.*, 2015). Intrusion Detection systems endeavours to identify computer attacks by probing into data records detected by processes on the same network (Jatuphum, *et al.*, 2015). An IDS senses intrusions by scrutinizing a network or system and analysing an audit stream taken from the network or system to look for evidences of mischievous behaviour (Lina, *et al.*, 2012). These attacks are usually divided into two categories, host-based attacks and network-based attacks (Lina, *et al.*, 2012). Host based attack detection practices usually use system call data from an audit activity that monitors all system calls made on behalf of each user on a specific machine (Hamdan, *et al.*, 2010). These audit activities normally run on each computer

system been monitored. Network-based attack detection procedures usually use network traffic data from a network packet sniffer such as tcp dump (Hamdan, *et al.,* 2010). Several computer networks, as well as the popular Ethernet (IEEE 802.3) network, employ a mutual channel for communication. Consequently, the packet sniffer only needs to share the same subnet with the computer systems been monitored (Denning,1986).

It is normal for every company or establishment to have security machinery in order to safeguard their data. One such security machinery is the Intrusion Detection System (IDS), which monitors and identifies intrusions, or anomalous events in computers or computer networks (Kavitha and Suresh, 2012). As a result of that, the IDS normally recognizes what represents "normal" event, and then an aberration of normalcy possibly will be an intrusion (Denning, 1986) or the IDS already have prototypes of suspicious events that signify that intrusions have taken place. These are then compared with present activities, to be able to conclude if there is an intrusion or not. The previous kind of detection is referred to as anomaly detection and the later misuse detection (Kim, Lee, and Kim, 2014). Intrusion Detection Systems (IDS) are required to detect events that portend danger to privacy, accessibility, and veracity of resources (Salma, *et al.,* 2015). The purpose of IDS is to discover intruders who lawfully go through the firewall unhindered (Kavitha and Suresh, 2012). Furthermore, insiders (intruders who have gained access to the firewalls) who are using their privilege to achieve ulterior motive can only be discovered by using an IDS (Kavitha and Suresh, 2012). There are basically two major kinds of IDS: Host-based Intrusion Detection System (Amrit and Manik, 2014) where the IDS dwell on a standalone host computer and monitor all the events for any malicious action (Firkhan and Yee, 2011). The second type is called the Network-based Intrusion Detection System (NIDS) which is usually installed on the network, and is intended to monitor network traffic (Umer, *et al.*, 2017). According to Manu Bijone (2016), the NIDS scans through the traffic from one packet to the other in real time, or close to real time, to endeavor to uncover intrusion patterns. There are two main approaches in devising IDS: anomaly and misuse based detection systems. In anomaly based detection methods, intrusions are identified as unusual behaviour that differ from the normal behaviour of the monitored system (Lata and Indu, 2013). The second approach for designing intrusion detection systems is misuse based detection. Attack patterns or signatures are identified and represented in such a way that the system can match these patterns with the log files or network traffic (Kavitha and Suresh, 2012). Most of the inductive learning based machine learning algorithms such as neural network (Sodiya, *et al.*, 2014) and (Raman, *et al.*, 2017) when applied to KDD 1999 Cup intrusion detection dataset usually resulted in awful performance for user-to-root and remote-to-local attack categories (Manu, 2016). Lin, *et al.*, (2015) applied k-NN and clustering for intrusion detection. There is therefore the need to explore other machine learning algorithms that can perform better than the ones that are already in use for intrusion detection (Manu, 2016). The classification of intrusion detection systems is as depicted in figure 1 below.

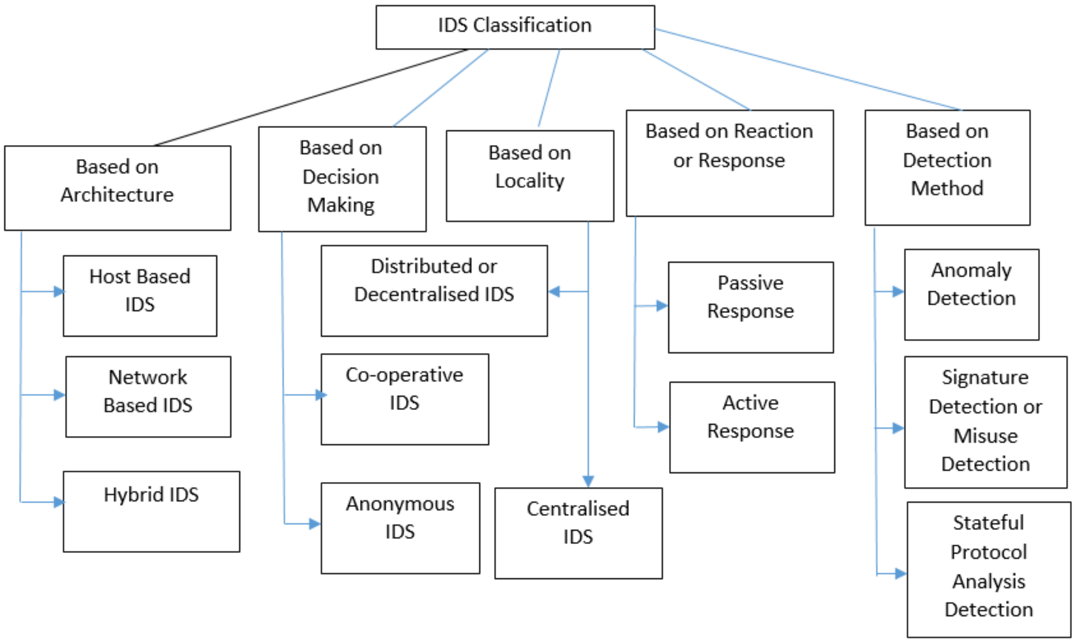

Figure 1: Classification of Intrusion Detection Systems based on used Technique (Source: Manu, 2016)

The objectives of this study is to evaluate the effectiveness of different machine learning techniques in detecting intrusion in a network by using the KDDTrain+.CSV dataset to analyse how each of these techniques can accurately detect different types of attacks: Probing attacks (information gathering attacks), Denial-of-Service (DoS) attacks (deny legitimate requests to a system), user-to-root (U2R) attacks (unauthorized access to local super-user or root), and remote-to-local (R2L) attacks (unauthorized local access from a remote machine). The figure 2 below shows the different types of attacks on a network.

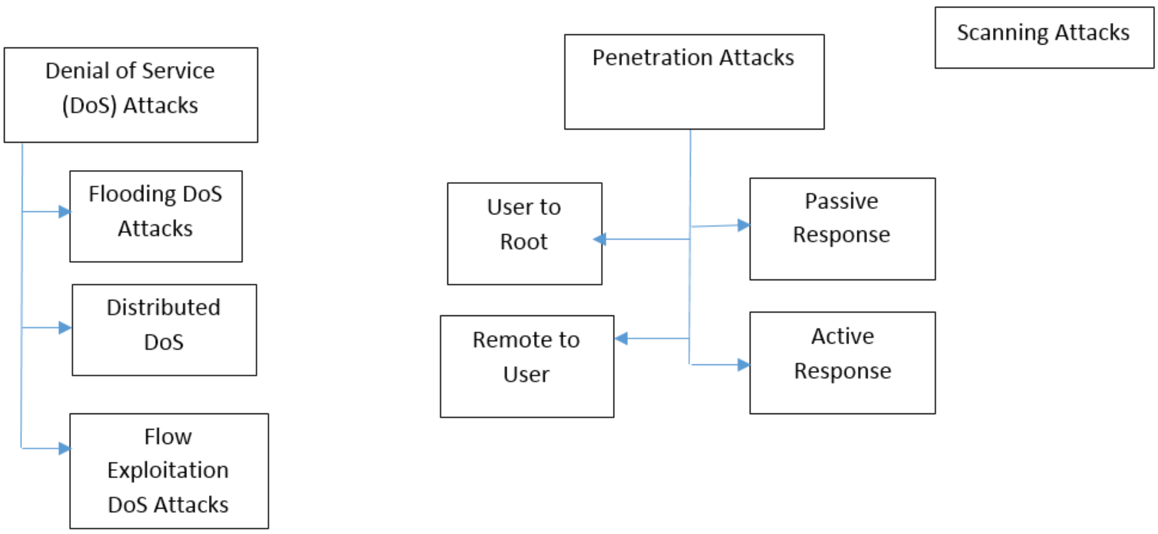

Figure 2: Types of Network Attacks (Source: Manu, 2016)

No new algorithm is proposed in this study. The contributions of this work are given as follows: review of some of the state of the art intrusion detection systems was done. Through this, we highlighted some interesting research gaps and research directions. A brief discussion of the different types of Intrusion detection systems and their classifications was also done. We did a study on six (6) machine learning techniques applied to intrusion detection and a review of literatures on intrusion detection over the period (2004-2017). This will expose researchers to some powerful machine learning algorithms whose effectiveness can be increased to cope with the complexity of security threats on computer networks. This can also serve as a springboard for naïve researchers and students who want to pursue research in the field of cyber security or machine learning.

2. Overview of Intrusion Detection

2.1 Simulators for Intrusion Detection

Intrusion traffic can be simulated in two ways, one is to simulate the intrusions in real environments, and the other is to simulate the intrusions in an experimental environment. In the first method, real machines in a much smaller scaled network communicate with each other. Each of them the machines plays different roles of attackers and victims. A classic example of data that can be used for intrusion detection is the KDD 99 dataset and the NSL-KDD benchmark dataset. The second approach involves a simulation software which creates network environment in perhaps, a single computer, and each network entity is simulated, and so does the traffic. Some of the most popular network simulation software includes PCP, NS2, and OPNET (2012) used in Razak, *et al.*, (2002).

2.2 Misuse Intrusion

This approach involves detection of intrusions by monitoring network traffic in the quest to locate direct matches of known patterns of attack (known as signatures or rules). This type of detection is also occasionally referred to as "signature based detection". A popular type of misuse detection that is utilized in commercial IDS where each pattern of events that resembles a different signature are classified as attack. However, there are more sophisticated approaches called state based analysis that can leverage a single signature to detect a group of attacks. A drawback of this technique is its ability to only detect intrusions that tally to an already specified signature or rules (Bace, 2000). There is need to relentlessly update the set of signatures to keep up with the new attacks. One merit of these approach is that the rate at which they raise false alarm is very low (Bace, 2000). Normally for the IDS to have the entire set of user activity, it processes the list of records obtained from the operating system for a particular duration of time (Bace, 2000; Crosbie and Spafford, 1995). After that, the IDS performs analysis of the current activity, using a rule base system, statistics, or a corresponding heuristic, in order to determine the possible occurrence of abnormality or intrusion.

2.3 Anomaly Detection

In this approach, the system defines the expected behaviour of the network in advance (Gunupudi, *et al.*, 2017). The profile of normal behaviour is built using techniques that include statistical methods, association rules and neural networks (Ghosh *et al.*, 1999) and (Przemysław, *et al.*, 2015). Any significant deviations from this expected behaviour are reported as possible attacks (Patcha and Park, 2007). The measures and techniques used in anomaly detection include: Firstly, threshold detection where certain attributes of user behaviour are expressed in terms of counts, with some level established as permissible (Gunupudi, *et al.*, 2017). Some examples of these attributes include

number of files accessed by a user in a given period, the number of failed attempts to login to the system, the amount of CPU utilized by a process. Secondly, statistical measures in which case the distribution of profiled attributes is assumed to fit a pattern. And other techniques which include data mining, neural networks, genetic algorithms and immune system models. In principle, the primary advantage of anomaly based detection is the ability to detect novel attacks for which signatures have not been defined yet. However, in practice, this is difficult to achieve because it is hard to obtain accurate and comprehensive profiles of normal behaviour. This makes an anomaly detection system generate too many false alarms and it can be very time consuming and labour intensive to sift through this data (Waskita, *et al.*, 2016). Most of the network-based intrusion detection techniques, however, are anomaly-based. A quite frequently used method by network anomaly-based IDS is to set up a statistical model of normal network traffic. A deviation from the model will be marked as suspicious (Gunupudi, *et al.*, 2017). Those statistical models are built from different perspectives of the traffic analysis (Waskita, *et al.*, 2014). Current network anomaly systems such as NIDES by Anderson (1995), EMERALD (2001), ADAM by Barbar *et al.*, (2001), and SPADE (2002) belong to this category.

3. Machine Learning Techniques for Intrusion Detection

Recently machine learning (ML) algorithms have become very popular as an effective technique for detecting the activities of intruders in a network. The appropriateness of machine learning algorithms for intrusion detection is due to the fact that they do not depend on hand-coded rules that are prone to the continually changing characteristics of anomalies in network. ML methods have the capacity to obtain information and learn from traffic packets provided, and then use the acquired information to classify new incoming packets. According to Mitchell (1997), ML algorithms have the ability to perform better based on their experience. In this section we will discuss some of the most popular machine learning techniques that have been applied to intrusion detection.

3.1 Naïve Bayes (NB)

The Bayesian classification is an example of supervised learning technique and at the same time a statistical technique for classification. It acts as a fundamental probabilistic model and let us seize ambiguity about the model in an ethical way by influencing the probabilities of the results. It is used to solve analytical and predictive problems. Bayesian classification is named after Thomas Bayes (1702 - 1761), who proposed the algorithm. The classification offers practical learning algorithms and previous knowledge and experimental data can be merged. Bayesian classification provides an advantageous perspective for understanding and evaluating many learning algorithms. A Naive Bayes classifier is a straightforward probabilistic classifier that is founded on using Bayes theorem with sound assumptions that are independent in nature (Biju and Wendy, (2009); Wu and Deng, (2008)).

$$\text{Bayes Theorem: Prob (B given A)} = \text{Prob (A and B)/Prob (A)} \qquad (1)$$

NB have been widely accepted as a simple and computationally efficient algorithm with satisfactory performances in solving real-world problems.

3.2 Multi-Layer Perceptron (MLP) Neural Networks

Artificial Neural Networks are groups of simple processing units, which are interconnect and communicate with one another by means of a sizable number of weighted connections. Each of the units accepts input from the neighbouring units and external sources and calculates the output that is transmitted to other neighbours. The means for modifying the weights of the nodes is also made available. Neural networks are potent algorithm for solving any machine-learning problem that requires classification. They are extensively use in different of applications areas such as recommendation engines, computer vision, and dashboard customization (Edstrom, 2016). Below is a multi-layer perceptron algorithm which is a standard Neural Network algorithm. The multi-layer perceptron assists in locating a linear function of the attribute vector $f(x) = w^T x + b$ such that $f(x) > 0$ for vectors of one group (Awad and Foqaha, 2016), and $f(x) < 0$ for vectors of other group. Also, w = (w_1 $w_2, \ldots w_m$) are the weights of the function, and b is the supposed bias. The groups can be given the numbers +1 and -1, so search for a function $d(x)$ = sign ($w^T x + b$) is carried out. The perceptron learning begins by randomly selecting parameters (w_0,b_0) of the resolution and repeatedly bringing them up-to-date. A training sample (x,c) is selected at the nth iteration of the algorithm to the extent that the present decision function now group it as incorrect (i.e. sign ($w_n x + b_n$) $\neq c$). The rule below is used in updating the parameters (w_n,b_n):

$$w_{n+1} = w_n + cx \qquad b_{n+1} = b_n + c \tag{2}$$

The criteria for terminating the algorithm is that a decision function must be located which accurately categorizes all the training samples into different groups (Carpinteiro, 2006).

3.3 J48 Decision Tree

A Decision Tree (DT) is a type of classifier whose pattern looks like that of a tree structure.J48 is a modified, redistributed and freely available version of C4.5 decision tree algorithm. J48 is developed by studying data at the nodes which are used to examine the meaning of prevailing attributes. A tree model is produced by the decision tree through the use of only one feature at a time. The algorithm uses the value of the feature to rearrange the dataset. And proceed to search for the areas of the dataset that obviously have one class and indicate those areas as leaves. The rest of the areas that contain classes that are more than one, the algorithm selects alternative features and maintain the dividing process with just the number of occurrences in such areas pending the time that the leaves are completely created or there is absence of feature that can be utilized to create at least one leave varied in the conflicted areas. The outcome of the classification or result generated is represented by a leaf node (Christina, *et al.*, 2010).

3.4 Logistic Model Tree Induction (LMT)

LMT is a type of decision tree that have logistic regression models at the leaves. This classifier has shown its capacity to generate a very high degree of accuracy in different research fields. The main shortcoming of this approach is its high computational complexity. The prediction of a model is created by organizing the tree down to the leaf and applying the logistic prediction model related to such leaf. The strength of the logistic model is that it is simple to understand and translate when compared to C 4.5 trees. Moreover, it has been proven that trees produced by LMT have a smaller size compared to those created by C 4.5 induction. The authors Chakraborty and Mondal (2012) explained in their paper that there is a decrease in training time needed to create the Logistic model tree compared to Naïve Bayes classifier and also produces better result than Naïve Bayes classifier.

3.5 Random Forests (RF)

Random forest (RF) belong to the class of ensemble learning method and regression approach that are suitable for solving problems that deals with classifying data into classes (Akinyelu and Adewumi, 2016). Random forest was first proposed by (Breiman and Cutler, 2007). The algorithm uses decision trees for solving classification tasks. At the training stage, some of decision trees are created by the program writer. These decision trees are subsequently utilised for the task of predicting the group; this is accomplished by taking into account the selected groups of every distinct trees and the group that have the highest number of vote is taken as the result. RF technique is gaining more popularity these days and it has find application in different fields and in literature it has been used to provide solution to analogous problem (Fette, *et al.*, 2007), (Koprinska, *et al.*, 2007) and (Whittaker, *et al.*, 2010). Some of the strength of Random forests is that it usually have lesser classification error and superior f-scores compared to decision trees.

3.6 Support Vector Machine Classifiers

Support Vector Machines are supervised learning algorithms that have been proven to perform better than some other attendant learning algorithms (Feng, *et al.*, 2014). It has been used to solve both classification and regression problem. SVM has find application in solving quadratic programming problems that have inequality constraints and linear equality by differentiating different groups by means of a hyperplane take full advantage of the boundary (Torabi, *et al.*, 2015). In summary, SVM watches for a linearly separable hyperplane, or a decision borderline where members of one class are separated from the other. The task of classification is considered accomplished wherever the hyperplane is present. In the absence of the hyperplane a nonlinear representation is used by SVM to convert the training data to a higher dimension after which it seeks for the best linear dividing hyperplane. The combination of a suitable nonlinear mapping to an adequately high dimension makes the division of data from two classes by a hyperplane possible. The hyperplane is located by SVM algorithm through the use of support vectors and borderlines.

4. Methodology

In this section, we evaluated the performance of the different machine learning approaches discussed in the previous section. The NSL-KDD dataset was used as the benchmark dataset. For fairness and purpose of easy comparison of all the algorithms, all machine learning algorithms investigated in this paper were simulated using WEKA 3.8 (Hall *et al.*, 2009). All experiments were conducted on a machine with a AMD A 10-7300 Radeon R6, 10 Compute Cores 4C+6G, 1.90 GHz, 8.00GB of RAM.

4.1 Data Preprocessing

Data in the real world is in their raw state. This means that they are incomplete (lacking attribute values, lacking certain attributes of interest, or containing only aggregate data. Many of the data are also noisy, which means that they contain errors or outliers. Many real world data are inconsistent (containing discrepancies in codes or names such as discrepancy between duplicate records. Incomplete data may come from “not applicable” data value when collected. There can also be different considerations between the time when the data was collected and when it was analyzed. Noisy data (incorrect values) may come from faulty data collection instruments Human or computer error such as data entry errors in data transmission etc.

Inconsistent data may be as a result of integration of different data sources from different customer data, like addresses, telephone numbers; misspelling of words etc. Data Preprocessing is very important because where quality data is lacking, there can be no quality detection result. And detecting an intrusion in the network might be difficult. Quality decisions must be based on quality data e.g., duplicate or missing data may cause incorrect or even misleading statistics. Data warehouse needs consistent integration of quality data. Data extraction, cleaning, and transformation comprise the majority of the work of building a data warehouse. Major tasks in data preprocessing is data cleansing. It involves filling in missing values in smooth noisy data, identify or remove outliers, and resolve inconsistencies data integration of multiple databases, data cubes, or files. Data transformation, normalization, and aggregate data reduction is another task carried out during data preprocessing. Data preprocessing obtains reduced representation in volume of data but produces the same or similar analytical results (restriction to useful values, and/or attributes only). Data discretization is part of data reduction but with particular importance, especially for numerical data.

4.2 Dataset Used

The performance of the different method used for this study was evaluated through experiments using the NSL-KDD (Tavallaee, *et al.*, 2009) dataset, which is an better version of Knowledge Discovery and Data Mining (1999) dataset (KDD'99). It was discovered during the analysis of the KDD'99 dataset (Tavallaee, *et al.*, 2009) that there was an intrinsic problem that have to do with numerous redundant instances present in the training and testing datasets, which have a significant negative impact on its performance, leading to poor assessment of the anomaly detection techniques. The KDD dataset has a very large number of duplicate records which need to be removed. To solve this problem, the NSL-KDD dataset was proposed by Tavallaee, *et al.* (2009), eliminating all redundant instances, and restructuring the dataset; which make available a more accurate and efficient evaluation of the different learning techniques. In this research, the evaluation dataset was organized by amending the NSL-KDD as follows:

4.1.1. Training Data

The training dataset is prearranged by the KDDTrain+.TXT document in the NSL-KDD dataset. The whole NSL-KDD training set is made up of 125,973 instances. KDDTrain+.TXT, the instances were removed.

4.1.2. Testing Data

The anonymous attack data set was arranged by altering the KDDTest+.TXT. Since the KDDTest+.TXT is comprised of instances that are comparable to those present inKDDTrain+.TXT, the instances were eliminated.

4.2. Performance Evaluation Measures

To evaluate the performance of the proposed method, six widely used performance metrics were used. They are the Accuracy (ACC), True Positive Rate (TPR), False Negative Rate (FNR), Precision (PR), F-measure and Receiver Operating Characteristic (ROC) Curve. The performance metrics was applied and calculated, using the confusion matrix given in Table 1. They show the possible results of classification.

Table 1. Confusion Matrix

Classified	**Actual**	
	Negative Class (Normal)	Positive Class (Anomaly)
Negative Class (Normal)	True negative (TN)	False positive (FP)
Positive Class (Anomaly)	False negative (FN)	True positive (TP)

These matrices are defined in the following equations below. They are denoted using the formula below:

$$TPR(Recall) = \frac{TP}{TP+FN} \quad (3)$$

$$PR(Precision) = \frac{TP}{TP+FP} \quad (4)$$

$$FPR = \frac{FP}{FP+TN} \quad (5)$$

$$FNR = \frac{FN}{TP+FN} \quad (6)$$

$$F_{-measure} = \frac{(1+Beta_2)*recall*precision}{Recall+Beta_2*Precision+Recall} \quad (7)$$

An ideal classifier should not produce False Positive (FP) and False Negative (FN) statistical errors. To evaluate non-ideal classifiers, one could measure proportion of correct assessments to all assessments called Accuracy (ACC).The share of benign activities reported as anomalous is known as False Positive Rate (FPR) and the share of anomalies missed by the detector is referred to as False Negative Rate (FNR). Usage of Precision (proportion of correctly reported anomalies) and Recall (share of correctly reported anomalies compared to the total number of anomalies) is another option. Recall also known as TPR is a basic factor that can determine the effectiveness of a classifier. It can be defined as the comparative number of intrusion that the classifier succeeded in detecting and preventing from entering the network. Precision can also described as the reliability of the filter which is calculated by the number of the intrusion classified by the intrusion detection technique as anomaly but are definitely normal. Based on these measures some tools like Receiver Operating Characteristics (ROC) and Precision vs Recall (PR) are typically used (Wu *et al.*, 2013). F-measure make use of a parameter that enables a compromise to be reached concerning recall and precision. F1 is the traditional F-measure that is commonly used and it presents uniform weight to recall and precision. In a situation where we have 0<*Beta*<1, it gives more importance to the precision while when we have *Beta*>1, it gives more importance to the recall. It is glaring that F-measure (F1-score) is an exclusive case of weighted F-measure when *Beta*=1.

4.3. Results and Discussion

As stated before, the goal of this research work is to evaluate the effectiveness of different machine learning algorithms discussed in this paper. The effectiveness can be evaluated using F-measure, TPR, PR, ACC, FPR and ROC. The efficiency is measured by the mean execution time in the training and testing phase. For a better understanding the performances of the different techniques, all the algorithms discussed in this paper were compared. The experimental results are in the table 2 below:

Table 2: Performance Comparison of different machine learning techniques

Technique	TPR	FPR	Precision	F-measure	ACC	OC Area
SVM	0.945	0.0053	0.962	0.964	0.960	0.947
LMT	0.952	0.0032	0.989	0.991	0.994	0.986
MLP	0.955	0.0368	0.976	0.989	0.980	0.978
J48	0.941	0.789	0.867	0.869	0.870	0.791
NB	0.952	0.632	0.868	0.889	0.895	0.810
RFs	0.952	0.737	0.745	0.747	0.755	0.755

The results presented in Table 2 compare the overall (detection and classification) performance of the six methods been investigated. Overall, the results shows that LMT method achieves better performance in terms of TPR, Precision, F-measure, ACC, and ROC (as Figure 3(a)-(f) shows the ROC curves of the different techniques been compared).

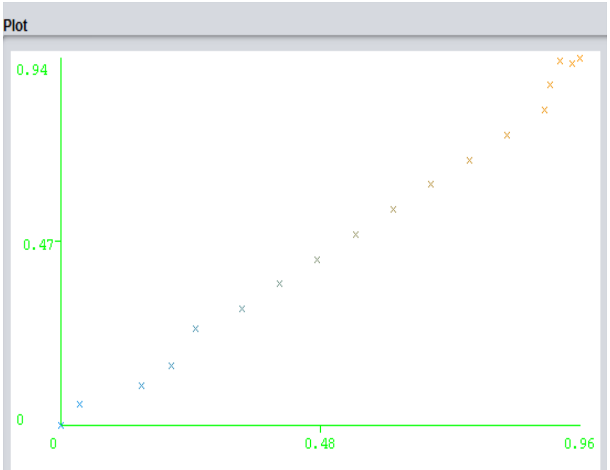

Figure 3(a): ROC curve for SVM

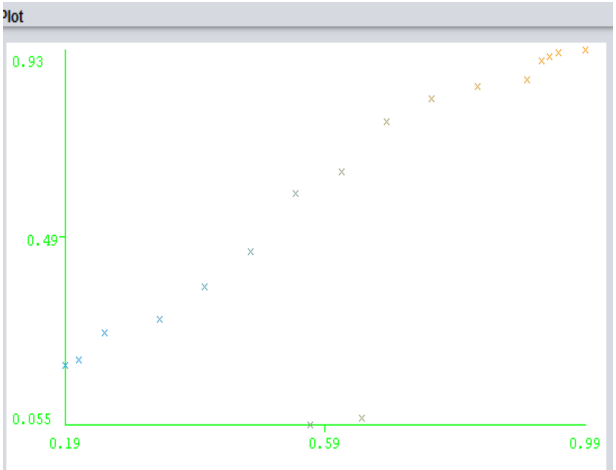

Figure 3(b): ROC curve for LMT

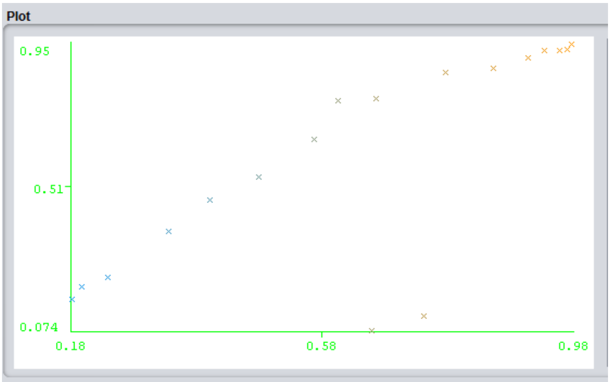

Figure 3(c): ROC curve for MLP

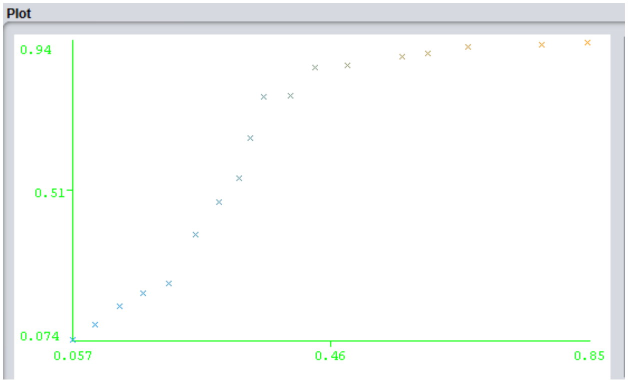

Figure 3(d): ROC curve for J48

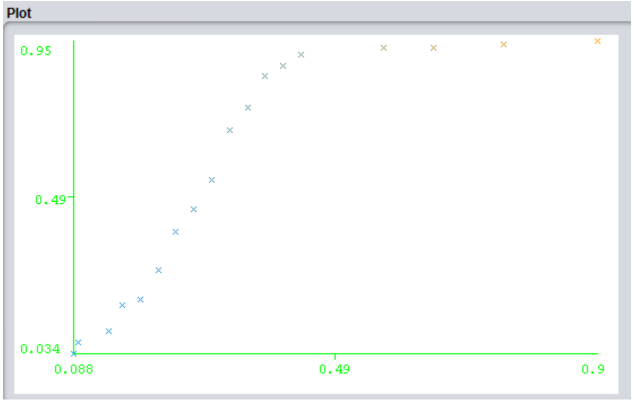

Figure 3(e): ROC curve for NB

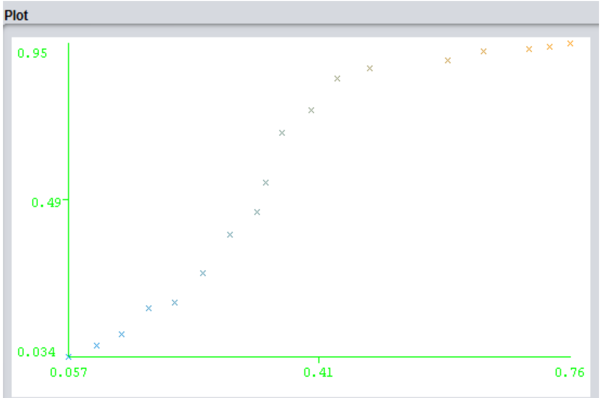

Figure 3(f): ROC curve for RFs

In order to evaluate the effectiveness of the various machine learning techniques under investigation, we drew a comparison among them. In terms of the classification accuracy (ACC), the results show that the LMT is 24% better than RFs, 3.4% better than the SVM algorithm, 10.5% better then NB, and 1.4% better than MLP. The MLP is 11% better in term of accuracy than J48 and 22.95% better than RFs. The poorest performance in term of accuracy of classification is from RFs (75.5%) followed by J48 (87%).Furthermore, the NB is 13.94% lower than the LMT, and the SVM is 2.81% lower than the LMT, in terms of Precision (PR).In terms of FPR, the LMT is 24.4% better than the RFs, 12.2% better than the J48, 3.36% better than the MLP, and 0.21% better than the SVM. An intrusion detection classifier with a lower FPR is a better classifier. In terms of F-measure, the LMT is 12.2% better than the J48, 10.2% better than the NB, 1.1% better than the MLP, and 2.7% better than the SVM. The superior performance of the LMT is because the algorithm combines the logistic regression models with tree induction, and so is an equivalent of model trees for solving classification problems. According to Niels, *et al.,* (2004), LMT shows little prejudice but frequently high variance: it explores a less bounded space of models, permitting it to capture nonlinear patterns in the data, but reduces its constituency thereby making it to be susceptible to over fitting. Perlich *et al.,* (2003) have shown that their comparative performance rests on the magnitude and the attributes of the dataset used (for instance, the signal-to-noise ratio).

5. Conclusion

There is an increasing number of intruders that have constituted security threat to networks and computer systems. This has necessitated the development of more sophisticated security systems that are far more effective than the conventional firewall systems. This paper investigates the performances of six (6) machine learning techniques in relation to how they can effectively handle different types of attack on networks using NSL-KDD dataset. All experiments were conducted on Weka 3.8. The strengths and shortcomings of the approaches to intrusion detection called Intrusion Detection System (IDS) and different classification of IDS were explained. As intrusion detection systems are becoming more sophisticated, the data generated by software and the techniques of the attackers are also getting more complicated on daily basis thereby making it very difficult to differentiate between legitimate and intrusion events in a system. Anytime an IDS wrongly classifies an action as a likely intrusion it will generate a false alarm, which is also known as a false positive. This can result to IDS erroneously blocking a valid user's IP address while leaving an intruder unchecked. There is therefore the need to develop more effective IDS. The overall conclusions for

our investigation is that it is possible to detect intrusion attacks based on anomalous patterns in network with machine learning based techniques. Machine learning classifiers generally have a high pedigree of effectiveness in identifying intrusion activities on a network. The results of the experiments conducted in this study shows that LMT performed best. It is the most efficient classifier (among famous classifiers used in Weka 3.8) based on our investigation. RFs turned out to be worse both in Accuracy (75.5%), FPR (73.7%), Precision (74.5), F-measure (74.7%) and as well as weighted ROC curves (75.5%). In the future, effort will be directed towards the design and implementation of more effective and efficient machine learning techniques that will have a better performance than all the algorithms discussed in this paper.

Reference

Akinyelu, AA. and Adewumi, AO. 2016. Classification of Phishing Email Using Random Forest Machine Learning Technique. Journal of Applied Mathematics, 6, Article ID 425731, Retrieved on July 12, 2017 from http://dx.doi.org/10.1155/2014/425731

Amrit, PS. and Manik, DS. 2014. Analysis of Host – Based and Network - Based Intrusion Detection System, I.J. Computer Network and Information Security, Vol. 8, pp. 41–47. DOI: 10.5815/ijcnis.2014.08.06

Anderson, D. 1995. Detecting unusual program behavior using the statistical component of the Next-generation Intrusion Detection Expert System (NIDES), Computer Science Laboratory SRI-CSL 95-06 May 1995.

Awad, M. and Foqaha, M. 2016. Email Spam Classification Using Hybrid Approach of RBF Neural Network and Particle Swarm Optimization. International Journal of Network Security and its Applications, Vol. 8, no. 4.

Bace, RG. 2000. Intrusion Detection. MacMillan Technical Publishing, USA.

Barbar, D., Wu, N. and Jajodia, S. 2001. ADAM: Dectecting Intrusions by Data mining", Proceedings of the 2001 IEEE, workshop on information Assurance and Security.

Biju, I. and Wendy, JJ. 2009. Implementing spam detection using Bayesian and porter stemmer keyword stripping approaches. In TENCON 2009-2009 IEEE Region 10 Conference, pp. 1-5.

Breiman, L. and Cutler, A. 2007. Random forests-classification description, Department of Statistics Homepage. http://www.stat.berkeley.edu/~breiman/RandomForests /cchome.htm.

Carpinteiro, OAS., Lima, I., Assis, JMC., de Souza, ACZ., Moreira, EM. and Pinheiro, CAM. 2006. A neural model in anti-spam systems., Lecture notes in computer science. Berlin, Springer.

Chakraborty, S. and Mondal, B. 2012. Spam Mail Filtering Technique using Different Decision Tree Classifiers through Data Mining Approach - A Comparative Performance Analysis. International Journal of Computer Applications (0975 – 888), vol. 47, no. 16, pp. 26-31.

Christina, V., Karpagavalli, S. and Suganya, G. 2010. Email Spam Filtering using Supervised Machine Learning Techniques, International Journal on Computer Science and Engineering, Vol. 02, no. 09, pp. 3126-3129.

Crosbie, M. and Spafford, G. 1995. Applying genetic programming to intrusion detection. In Papers from the 1995 AAAI Fall Symposium, pp. 1–8.

Denning, DE. 1986. An intrusion-detection model. In Proceedings of the 1986 IEEE Symposium on Security and Privacy, pp. 118–131.

EMERALD. 2001. Intrusion Detection System, Retrieved on December 15th, 2016 from http://www.sdl.sri.com/projects/emerald.

Feng, W., Zhang, Q., Hu, G. and Huang, JX. 2014. Mining network data for intrusion detection through combining SVMs with ant colony networks. Future Generation Computer Systems, Volume 37, July 2014, Pages 127-140. https://doi.org/10.1016/j.future.2013.06.027.

Fette, I., Sadeh, N. and Tomasic, A. 2007. Learning to detect phishing emails, in Proceedings of the 16th International World Wide Web Conference (WWW '07), Alberta, Canada, May 2007, pp. 649–656,

Firkhan, ABHA. and Yee, YL. 2011. Development of Host Based Intrusion Detection System for Log Files. 2011 IEEE Symposium on Business, Engineering and Industrial Applications (ISBEIA), Langkawi, Malaysia, Page 281 – 285.

Ghosh, AK., Schwartzbard, A. and Schatz, M.1999. Learning Program Behavior Profiles for Intrusion Detection, 1st USENIX Workshop on Intrusion Detection and Network Monitoring.

Gunupudi, RK., Nimmala, M., Gugulothu, N. and Gali, SR. 2017. CLAPP: A self-constructing feature clustering approach for anomaly detection, Future Generation Computer Systems, volume 74, pp. 417-429, https://doi.org/10.1016/j.future.2016.12.040.

Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P. and Witten, IH. 2009. The WEKA data mining software: an update", ACM SIGKDD explorations newsletter, vol. 11, no. 1, pp. 10-18.

Hamdan, O., Alanazi, RMN., Zaidan, BB. and Zaidan, AA. 2010. Intrusion Detection System: Overview. Journal of Computing, vol. 2, issue 2, HTTPS://SITES.GOOGLE.COM/SITE/ JOURNALOFCOMPUTING/

Jatuphum, J., Ngamnij, A., Somjit, A. and Saiyan, S. 2015. A Novel Lightweight Hybrid Intrusion Detection Method Using a Combination of Data Mining Techniques. International Journal of Security and Its Applications, vol. 9, no. 4, pp. 91-106. http://dx.doi.org/10.14257/ijsia.2015.9.4.10

Kavitha, C. and Suresh, M. 2012. Processing Massive Data Streams to Achieve Anomaly Intrusion Prevention. Fourth International Conference on Conference: Computational Intelligence and Communication Networks (CICN), 2012 DOI: 10.1109/CICN.2012.167.

KDD'99 dataset, The UCI KDD Archive Information and Computer Science University of California, Irvine, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

Kim, G., Lee, S. and Kim, S. 2014. A novel hybrid intrusion detection method integrating anomaly detection with misuse detection. Expert Systems with Applications, Volume 41, Issue 4, Part 2, pp. 1690-1700. https://doi.org/10.1016/j.eswa.2013.08.066.

Koprinska, I., Poon, J., Clark, J. and Chan, J. 2007. Learning to classify e-mail, Information Sciences, Vol. 177, no. 10, pp. 2167–2187.

Lata, and Kashyap, I. 2013. Study and Analysis of Network based Intrusion Detection System. International Journal of Advanced Research in Computer and Communication Engineering, Vol. 2, issue 5, pp. 2032 – 2038. Retrieved from https://www.ijarcce.com/upload/2013/may/17-lata%20bhardwaj%20-%20Study%20and%20analysis%20of%20NETWORK%20BASED%20intrusion%20detection%20system.pdf

Liao, HJ., Richard, CH., Lina, LYC. and Tung, KY. 2013. Intrusion detection system: A comprehensive review. Journal of Network and Computer Applications, volume 36, issue 1, pages 16-24. https://doi.org/10.1016/j.jnca.2012.09.004.

Lin, WC., Ke, SW. and Tsai, CF. 2015. CANN: An intrusion detection system based on combining cluster centers and nearest neighbors. Knowledge-Based Systems, Volume 78, April 2015, Pages 13-21. https://doi.org/10.1016/j.knosys.2015.01.009.

Lina, SW., Ying, KC., Lee, CY. and Lee, ZJ. 2012. An intelligent algorithm with feature selection and decision rules applied to anomaly intrusion detection. Applied Soft Computing, vol. 12, pp. 3285–3290. http://dx.doi.org/10.1016/j.asoc.2012.05.004.

Manu B. 2016. A Survey on Secure Network: Intrusion Detection & Prevention Approaches. American Journal of Information Systems, vol. 4, no. 3, pp. 69-88. DOI:10.12691/ajis-4-3-2.

Mitchell, TM. 1997. Machine learning (1st ed.). McGraw-Hill.

Niels L., Mark, H. and Eibe, F. 2004. Logistic Model Trees. Kluwer Academic Publishers, vol. 14, no. 21, pp. 1-54. Available at https://www.cs.waikato.ac.nz/ml/publications/2005/LMT.pdf

NS2, 2005. The Network Simulator, ns-2 http://www.isi.edu/nsnam/ns/

OPNET, 2012. Retrieved from http://www.opnet.com.

Patcha, A. and Park, JM. 2007. An overview of anomaly detection techniques: Existing solutions and latest technological trends. Computer Networks, 51.

Perlich, CF., Provost, and Simonoff, J. 2003. Tree Inductions vs. Logistic Regression: A Learning-curve Analysis. Journal of Machine Learning Research, Vol. 4, pp. 211–255.

Przemysław, B., Bartosz, J. and Marcin, S. 2015. An Entropy-Based Network Anomaly Detection Method. Entropy, vol. 17, pp. 2367-2408. doi:10.3390/e17042367.

Raman, MRG., Somu, N., Kirthivasan, K. and Sriram, VSS. 2017. A Hypergraph and Arithmetic Residue-based Probabilistic Neural Network for classification in Intrusion Detection Systems. Neural Networks, pp. 89-97. https://doi.org/10.1016/j.neunet.2017.01.012.

Razak, S., Zhou, M. and Lang, S. 2002. Network Intrusion Simulation Using OPNET. Proceedings of OPNETWORKS, 2002.

Salma, E., Alberto, F., Abdullah, B., Saleh, A. and Francisco, H. 2015. On the combination of genetic fuzzy systems and pairwise learning for improving detection rates on Intrusion Detection

Systems, Expert Systems with Applications, Vol. 42, issue 1, pp. 193-202. https://doi.org/10.1016/j.eswa.2014.08.002.

Sodiya, AS., Ojesanmi, OA. and Akinola OC. 2014. Neural Network based Intrusion Detection Systems, International Journal of Computer Applications (0975 – 8887), Vol. 106, No. 18, pp. 19-24.

SPADE. 2002. Silicon Defense, http://www.silicondefense.com/software/spice/

Tavallaee, M., Bagheri, E., Lu, W. and Ghorbani, AA. 2009. A detailed analysis of the KDD CUP 99 data set", Proceedings of Proceedings of the Second IEEE international conference on Computational intelligence for security and defense applications.

Torabi, ZS., Nadimi-Shahraki, MH., and Nabiollahi, A. 2015. Efficient Support Vector Machines for Spam Detection: A Survey. (IJCSIS) International Journal of Computer Science and Information Security, Vol. 13, No. 1, pp. 11-28.

Umer, MF., Sher, M. and Bi, Y. 2017. Flow-based intrusion detection: Techniques and challenges Computers and Security, Vol. 70, Pages 238-254, https://doi.org/10.1016/j.cose.2017.05.009.

Waskita, AA., Suhartanto, H. and Handoko, LT. 2016. A performance study of anomaly detection using entropy method. Proceeding - 2016 International Conference on Computer, Control, Informatics and its Applications: Recent Progress in Computer, Control, and Informatics for Data Science, IC3INA 2016.

Waskita, AA., Suhartantoy, H., Persadhazy, PD. and Handoko, LT. 2014. A simple statistical analysis approach for Intrusion Detection System. arXiv. Retrieved on July 15, 2017 from DOI: 10.1109/SPC.2013.6735130.

Whittaker, C., Ryner, B. and Nazif, M. 2010. Large-scale automatic classification of phishing pages, in Proceedings of the 17th Annual Network & Distributed System Security Symposium (NDSS ’10), The Internet Society, San Diego, Calif, USA.

Wu, J., and Deng, T. 2008. Research in anti-spam method based on bayesian filtering. In Pacific-Asia Workshop Computational Intelligence and Industrial Application, 2008. PACIIA ’08., Vol. 2, pp. 887-891.

Wu, Y., Cai, S., Yang, S., Zheng, F. and Xiang, N. 2013. Classification of Knee Joint Vibration Signals Using Bivariate Feature Distribution Estimation and Maximal Posterior Probability Decision Criterion. Entropy 2013, Vol. 15, pp. 1375–1387.